Efficient Location Training Protocols for Heterogeneous Sensor and Actor Networks

F. Barsi, A.A. Bertossi, C. Lavault, A. Navarra, S. Olariu, M.C. Pinotti, and V. Ravelomanana

***Abstract*— In this work we consider a large-scale geographic area populated by tiny sensors and some more powerful devices called actors, authorized to organize the sensors in their vicinity into short-lived, *actor-centric* sensor networks. The tiny sensors run on miniature non-rechargeable batteries, are anonymous and are unaware of their location. The sensors differ in their ability to dynamically alter their sleep times. Indeed, the *periodic* sensors have sleep periods of predefined lengths, established at fabrication time; by contrast, the *free* sensors can dynamically alter their sleep periods, under program control.**

The main contribution of this work is to propose an energy-efficient location training protocol for heterogeneous actor-centric sensor networks where the sensors acquire coarse-grain location awareness with respect to the actor in their vicinity. Our analytical analysis, confirmed by experimental evaluation, show that the proposed protocol outperforms the best previously-known location training protocols in terms of the number of sleep/awake transitions, overall sensor awake time, and energy consumption.

***Index Terms*— Sensor and actor networks, heterogeneous sensors, coarse-grain localization, location training protocols, localization protocols**

I. Introduction

We assume a large-scale deployment of heterogeneous micro-sensors, each perhaps no larger than a dime, and possessing only limited functionality, along with more powerful devices, called *actors*. The actors are authorized to organize the sensors in their vicinity into short-lived, *actor-centric* networks in support of a specific mission; when the mission terminates the networks are dissolved and the sensors return to their unorganized state [1], [2]. As an example, imagine a blind person that tries to cross the street in a sensor-instrumented city block. The blind person will organize the sensors in his/her immediate vicinity into a short-lived network whose stated goal is to help them chart a safe course to their destination. Once the blind person has been assisted, the sensor network is disbanded and the sensors return to their dormant state. This view, illustrated in Figure 1, is similar to that in [9], [15], [18] but differs from the prevalent contemporary view according to which sensor networks are deployed in support of a remote user that is querying the network and where the collected/aggregated data is sent to a remote site for final determination.

It is worth noting that in an *actor-centric* network the concept of globality needs to be redefined to mean small-scale spatial and temporal globality, the only viable form of non-local interaction. Indeed, no global aggregation or fusion of sensory data is performed because such operations do not scale well with the size of the deployment area. It has been argued that actor-centric sensor networks can detect trends and unexpected, coherent, and emergent behaviors and find immediate applications in environmental monitoring and homeland security [18], [24].

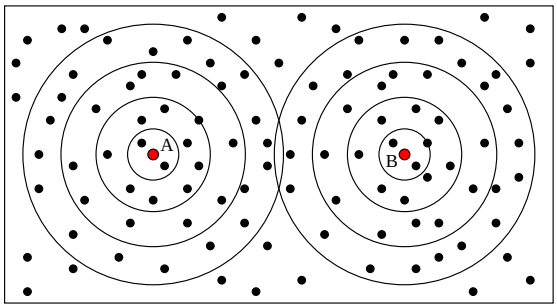

Fig. 1. *Illustrating two actor-centric networks.*

A number of applications benefit or even require that the sensory data collected by sensors be supplemented with exact location information, encouraging the development of location-aware and perhaps location-dependent communication protocols [16], [22], [25], [26]. However, in large-scale sensor deployments is it either infeasible or impractical to pre-engineer the position of individual sensors. The net effect of this state of affairs is that, as a rule, the sensors are initially unaware of their location: they must acquire this information post-deployment. In fact, in most of the existing literature, the sensors are assumed to have learned their geographic position [2], [3], [24]. The *location awareness problem* is for individual sensors to acquire location information either in absolute form (e.g. geographic coordinates) or relative to a reference point. The *localization problem* is for individual sensors to determine, as precisely as possible their geographic coordinates. One simple solution to the localization problem is to use GPS (global positioning system), where sensors receive signals from several satellites and decide their position directly. However, due to limitations in form factor, cost per unit and energy budget, tiny sensors are not expected to be GPS-enabled. Moreover, in many occluded environments including those inside buildings, hangars or warehouses GPS access is drastically curtailed [25].

F. Barsi, A. Navarra, and M.C. Pinotti are with the Department of Computer Science and Mathematics, University of Perugia, 06123 Perugia, Italy, {barsi,pinotti,navarra}@dmi.unipg.it

A.A. Bertossi is with the Department of Computer Science, University of Bologna, Mura Anteo Zamboni 7, 40127 Bologna, Italy, bertossi@cs.unibo.it

C. Lavault and V. Ravelomanana are with the Laboratoire d'Informatique de Paris-Nord, University of Paris 13, Villetaneuse, Paris, France, {Christian.Lavault,vlad}@lipn.univ-paris13.fr

S. Olariu is with the Department of Computer Science, Old Dominion University, Norfolk, VA 23529-0162, USA, olariu@cs.odu.edu

In many other applications, exact geographic location is not necessary: all that individual sensors need is *coarse-grain* location awareness. The task of acquiring such a coarse-grain location awareness, relative to a reference point, is referred to as *location training* (training, for short). There is an obvious trade-off: coarse-grain location awareness is lightweight but the resulting accuracy is only a rough approximation of the exact geographic coordinates. One can obtain this coarse-grain location awareness by a protocol that imposes a coordinate system onto the sensor network. Wadaa *et al.* [23] have shown that an interesting by-product of such a training protocol is that it provides a partitioning into clusters and a structured topology with natural communication paths. The resulting topology will make it simple to avoid collisions between transmissions of nodes in different clusters, between different paths and also between nodes on the same path. This is in contrast with the majority of papers that assume routing along spanning trees with frequent collisions.

Recently, a number of papers have studied location training protocols which impose a coordinate system by an actor, see [5], [18]. The typical mode of operation of an actor is to move towards the place where an event occurs, stationing there for a while so as to task the sensors in the circular field, centered at one of the actors (see Figure 2), for collecting data relevant to the mission at hand. In support of its mission, the actor is provided with a steady power supply and a radio interface for long distance communications. We assume that, in general, the actors are equipped with both isotropic and directional antennas. By means of the isotropic antenna, the actor is able to broadcast with variable-range R to reach all the sensors at distance at most R from the actor. Moreover, using the directional antenna, the actor can broadcast at full-range to all the sensors lying in a circular sector of arbitrary angle α with respect to the polar axis. When the actor transmits, all the awake sensors belonging to the area covered by the current transmission passively receive the actor's message. The potential of such an actor to train the sensors in its vicinity has been explored in [5], [6], [23] where location training protocols were presented which divide the sensor deployment area, consisting of a disk around the actor, into equiangular circular sectors and concentric coronas (i.e. areas between two concentric circles both centered at the actor). In this way, a large sensor deployment area of any shape is organized into several cooperating actor-centric subnetworks, one for each deployed actor (where sensors lying in the intersection ranges of many actors should refer to just one actor, choosing one of them).

As illustrated in Figure 2, after training, each sensor in a disk of radius R around the actor has acquired two coordinates, namely the corona and the sector to which it belongs. Notice that training provides for free a *clustering* of the sensors, where a cluster consists of all sensors having the same coordinates. After training, routing can be easily performed as follows. Cluster-to-actor messages are trivially routed inward within a single sector, while cluster-to-cluster messages can be routed following several paths, e.g., first along the sector of the sender to reach the corona of the receiver, and then within such a corona (clockwise or counterclockwise, depending on which is the shortest path) to reach also the sector of the receiver [23]. In addition, to help the actor locate an event that has occurred in the network, each sensor can add its coordinates to the sensed data before delivering the messages to the actor.

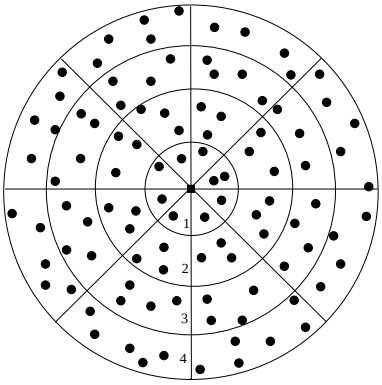

Fig. 2. *A trained actor-centric subnetwork*

The location training protocols studied thus far in the literature work on *homogeneous* sensors in terms of computing and communication capabilities as well as energy budget. By contrast, in this paper we look at training protocols that handle sensors with different capabilities. With sensors being deployed at various times by different infrastructure providers, heterogeneity is expected to be the norm in the sensor networks of the future.

We assume that the sensors run on miniature non-rechargeable batteries. When a sensor is awake, its CPU is active, along with its timer, and its radio is on; is sleep mode, the CPU is inactive, the radio interface is powered off, and only the timer is on. In order to promote longevity, the sensors spend most of the time in sleep mode, waking up for brief time periods only [1], [8], [19]. The heterogeneous actor-centric sensor networks considered hereafter involve two types of sensors: on the one hand, the *periodic* sensors have sleep periods predetermined fabrication-time that cannot be altered; on the other, the *free* sensors may alter their sleep periods dynamically under program control.

The main contribution of the present paper is to propose a novel location training protocol for actor-centric sensor networks with a heterogeneous sensor population. The behavior of the actor is based on linearly decreasing strength transmissions alternating with full strength transmissions. On the other hand, the sensors perform a binary search among the actor transmissions to locate their correct corona. Although the two types of sensors are driven by the same actor protocol, they locally act in different ways. The sensors are anonymous and indistinguishable to the actor. Each sensor starts the training task when it wakes up for the first time, without any initial explicit synchronization. It is assumed that, during the training task, both sensors and actor measure the time in slots, which are equal in both lengths and phase. However, every time a sensor receives a transmission from the actor, it can re-phase its own slot. This makes the protocol resilient to sensor clock drift.

The remainder of this work is organized as follows. Section II offers a succinct survey of localization protocols. Section

III introduces our actor-centric network model and details the actor and the sensor behavior of the proposed protocol. Section IV exhibits the worst-case performance analysis of the protocol, in terms of the number of sleep/awake transitions per sensor and thus in terms of energy consumed. Section V presents an experimental evaluation of the performance, tested on randomly generated instances, confirming the analytical results, and showing a much better average-case behavior. The performance is then compared with that of all the previous location training algorithms known for the periodic sensors, showing that the new protocol requires fewer sleep/awake transitions, and hence consumes much less energy per sensor. Finally, Section VI offers concluding remarks.

II. Related Work

The task of determining the exact location of sensors, referred to as *localization*, has been extensively studied in the literature [4], [13], [22], [24]. Since GPS is considered prohibitive, most solutions assume the existence of several GPS-enabled anchors. Localization algorithms can then be divided into two categories: *range-based* and *range-free* [12], [21]. In range-based algorithms, the sensors estimate their distance to anchors using some specialized hardware, and applying methods like triangulation or trilateration [4]. Other range-based algorithms use received signal strength, angle and/or time of arrival of signals, or difference of time of arrivals. Although range-based algorithms result in a fine-grain localization, all of them need special hardware which may not be feasible to provide at the sensor level. On the other hand, range-free algorithms do not use any special hardware but accept a less accurate localization. For example, in the range-free *centroid* algorithm, the sensors receive the anchor positions, and using this proximity information, a simple centroid model is applied to estimate the position of the listening nodes [7]. Other solutions use methods similar to distance vector routing to allow the nodes to find the number of hops from the anchors. Anchors flood their location throughout the network maintaining a running hop-count at each node along the way. Nodes calculate their position based on the received anchor locations, on the hop-count from the corresponding anchor, and on the average-distance per hop [17]. In [12], an iterative method is pursued to narrow down the position accuracy until a tolerable error in the positioning is reached. In practice, each sensor repeatedly chooses a triple of anchors from all audible anchors and tests whether it is inside the triangle formed by them, until all triples are exhausted or the required accuracy is achieved. At this point, the center of gravity of all of the triangles in which a node resides is assumed to be the sensor estimated position. Finally, some localization algorithms, called *proximity-based* algorithms, determine the node positions by making use of neighbor nodes, which act as anchors for other nodes [20].

The localization algorithms discussed so far assume that the anchor nodes are special nodes, mainly because they know their spatial coordinates, and that localization is performed as a primitive operation with all the sensors awake, thus ignoring energy saving achievable by utilizing the sleep-awake duty cycle of the sensors. Instead, several recent papers [6], [5], [23] have considered the localization problem in a network whose anchor nodes, called actors, are provided with special transmission capabilities and steady power supply (while do not necessarily need GPS receivers) and whose sensor nodes exploit sleep-awake duty cycles for saving energy. The main novelty of such papers is in using an actor to impose a discretized polar coordinate system and in combining for the first time localization and energy-efficient MAC protocols. In [6], [5], [23], localization is intended as the task of making each sensor able to acquire a coarse-grain location with respect to a given actor node and is referred to as location training. The process is centralized and uses only asymmetric broadcasts (from the actor to the sensors) without multihop communications among the sensors. The sensors deduce their coarse-grain location exploiting the information received by the actor without performing any local communication. In particular, the Flat corona training protocol and its variants, Flat$+$ and TwoLevel, proposed in [5], deal with a homogeneous network of periodic sensors. They are called *asynchronous* protocols because each periodic sensor learns the identity of the corona to which it belongs, regardless of the moment when it wakes up for the first time. On the other hand, the two protocols proposed in [6] deal with a homogeneous network of free sensors, and are fully synchronous.

In the Flat protocol, the actor cyclically repeats a transmission cycle which involves k broadcasts at successively decreasing transmission ranges, where k is the number of coronas. Each broadcast lasts for a slot and transmits a beacon equal to the identity of the outmost corona reached. On the other side, each sensor wakes up at random between the 0-th and the $(k-1)$-st time slot and starts listening to the actor for d time slots, that is, its awake period. Then, the sensor goes back to sleep for $L-d$ time slots, that is, its sleep period. Such a behavior is repeated until the sensor learns the identity of the corona c to which belongs, because it heard beacon c but not beacon $c-1$ although it knows that this latter beacon has been transmitted. The Flat$+$ extends the Flat protocol exploiting the fact that when a sensor hears a beacon c, it knows that it will also hear all the beacons greater than c. Similarly, when a sensor knows that a beacon c has been transmitted but not heard, it knows also that it cannot hear any beacon smaller than c. In contrast to the Flat protocol, the sensor now keeps track of beacons not yet transmitted during its awake periods, and thus it can look ahead and skip its next awake period. However, as proved in [5], the worst case performance remains the same as Flat. In a further improvement, called the *Two-Level* protocol, the actor follows a nesting approach in which the k coronas are viewed as k_1 macro-coronas of k_2 adjacent micro-coronas each. Each sensor is trained to learn first its macro-corona and then its micro-corona. Although Two-Level is the most efficient protocol known so far, it cannot reduce the number of sensor sleep/awake transitions below the square root of the number of transitions needed by the Flat protocol [5]. However, the actor behavior of Two-Level is designed ad hoc for periodic sensors and cannot handle free sensors.

The two training protocols presented in [6] assume that the sensors are free and synchronized to the master clock running

at the actor. The actor behavior in such protocols can be thought of as traversals of complete binary/d-ary trees, whose leaves represent coronas, whose node preorder/BFS numbers are related to the time slots, and whose node inorder/BFS numbers are related to the actor transmission ranges, respectively. By exploiting the fully synchronized model and by performing a distributed phase where the sensors that have already learned their corona inform those in their neighborhood, the protocols in [6] require a logarithmic number (in the number of coronas) of sensor sleep/awake transitions and achieve an optimal square root time (also in the number of coronas) for terminating the training task. However, the need of a strong synchronization between the actor and the sensors makes this protocol difficult to extend to periodic sensors.

This paper presents an asynchronous protocol, where the actor repeats a transmission cycle at decreasing transmission ranges, which can simultaneously train both free and periodic sensors. Our protocol improves over all the previously asynchronous protocols by reducing the number of sleep/awake transitions to a logarithmic number, thus matching the fastest synchronous protocol presented in [6].

III. The Binary Training Protocol

This section serves the dual purpose of specifying the details of our network model and that of presenting the training protocol. As a result of running the training protocol, each sensor will acquire the desired coarse-grain location awareness (namely, the identity of the corona and sector to which it belongs), regardless of its type and of the moment when the sensor wakes up for the first time.

A. Network Model and Problem Formulation

We restrict our attention to an actor-centric sensor network consisting of an actor and the set of sensors in a disk centered at the actor. For simplicity we assume that the disks covered by distinct actors are disjoint. This is the case in many practical applications [18].The actors have a steady power supply and a special radio interface for long distance communications. Time is ruled into slots. Both the sensors and the actors use identical, in phase, slots. If the slot maintained at a sensor drifts from that at the actor, the sensor can easily re-phase its slot every time it wakes up, as it will be shown in the protocol. The sensors operate subject to the following fundamental constraints:

- The sensors are *anonymous* – to assume the simplest sensor model, the sensors lack individually unique IDs;
- The sensors have a modest non-renewable energy budget;
- No sensor has global information about the network topology;
- All sensors can receive isotropic transmissions emanating from the actor;
- The sensors are *asynchronous* – they wake up for the first time according to their internal clock and do not engage in explicit synchronization protocol with the actor or other sensors.

We assume that the sensors come in two flavors:

- The *periodic* sensors alternate between sleep and awake periods, both of fixed length. The sensor sleep-awake cycle has a total length of L time slots, out of which the sensor is awake for $d \leq L$ slots. Periodic sensors may sleep for their entire cycle, skipping awake periods, as depicted in Figure 3;

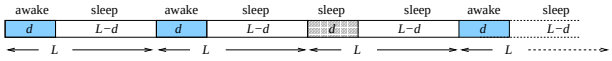

Fig. 3. *Illustrating the sleep-awake cycle of a periodic sensor. The darkest d slots represent a time interval in which the sensor was scheduled to be awake but decided to sleep instead.*

- The *free* sensors alternate between sleep periods, whose lengths depend on the executed protocol and can assume arbitrary values, and awake periods of fixed length d, as shown in Figure 4.

Fig. 4. *Illustrating a free sensor that alternates between awake periods of fixed length d and sleep intervals of arbitrary lengths.*

The training problem considered in this paper asks for imposing a coordinate system onto the sensor deployment area by establishing:

1. *Coronas*: The deployment area is covered by k coronas $C_0, C_1, \ldots, C_{k-1}$ determined by k concentric circles, centered at the actor, whose radii are $0 < r_0 < r_1 < \cdots < r_{k-1}$;
2. *Sectors*: The deployment area is ruled into h equiangular sectors $S_0, S_1, \ldots, S_{h-1}$, centered at the actor, each having a width of $\frac{2\pi}{h}$ radians;

where k and h are system parameters determined before the algorithm starts. The objective is for each sensor to acquire the identity of the corona and sector to which it belongs, while consuming as little energy as possible.

To avoid handling tedious and inconsequential details, we assume that all coronas and sectors have the same width, although this is not strictly required [19]. In a practical setting, the corona width might equal the sensor actor's transmission range, say r, and hence the (outer) radius r_i of corona C_i might be equal to $(i+1)r$. In such a case, then, the corona number plus one gives the number of hops needed for a sensor-to-actor communication. Moreover, a sector S_j might consist of the portion of the area between the two directional transmission angles $j\frac{2\pi}{h}$ and $(j+1)\frac{2\pi}{h}$. It is further assumed that the free and periodic sensors share the same awake period length d and that all periodic sensors share the same sleep period length $L-d$, and that both d and L are even. However, the protocol to be discussed can handle sensors with different sleep and awake parameters, where each single free sensor f has its own (even) awake period length d_f, and each single periodic sensor p has its own (even) awake and sleep periods of length d_p and $L_p - d_p$, respectively.

B. The Actor Behavior

Consider first the training of the coronas. The pseudocode of the actor behavior is given in Figure 6. The actor repeats

a cycle of $2k$ time slots and transmits in each slot a message, called a *beacon*, consisting of a corona identity. At time slots $2(k-1-i)$ and $2(k-1-i)+1$, with $i = k-1, \ldots, 0$, the actor broadcasts a *control-broadcast*, followed by a *data-broadcast*, using its isotropic antenna. Both broadcast the beacon i, the former at full power, reaching all the sensors, and the latter at a power level reaching only those sensors up to corona C_i. The actor transmission cycle, featured in Figure 5, is repeated for a time τ sufficient to accomplish the training protocol. An evaluation of τ will be given in Theorems IV.6 and IV.9 (for free and periodic sensors, respectively).

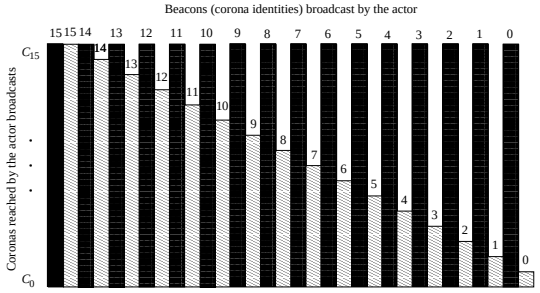

Fig. 5. *An actor transmission cycle of $2k$ time slots with $k = 16$. The actor alternates control-broadcasts at full power level (black) with data-broadcasts at decreasing power levels (gray), transmitting corona identities in decreasing order.*

The redundancy inherent in the control-broadcasts allows the sensors to hear the beacons transmitted in the data-broadcasts even when they are out of the data-broadcast ranges, and thus to acquire information about their distance from the actor. One reason for performing data-broadcasts in descending order of coronas is that the outer coronas, which contain more sensors than the inner ones, are reached first. Moreover, since for free sensors, as proved in Lemma IV.4, the inner coronas complete their training earlier than the outer coronas, a subnetwork connected to the actor grows and could start operating before the whole training task terminates.

The training of sectors is analogous to the training of coronas, except that now the actor broadcasts using the directional antenna a beacon consisting of a sector identity. The actor cyclically repeats a transmission cycle of $2h$ directional broadcasts with successively smaller angles. Specifically, at time slots $2(h-1-i)$ and $2(h-1-i)+1$, with $i = h-1, \ldots, 0$, the actor broadcasts at a full power level a control-broadcast and a data-broadcast both transmitting the same beacon. The actor uses in such two broadcasts proper angles of transmission so as to reach all the sensors lying in all the sectors and those up to sector S_i, respectively. Since sector training is the same as corona training once the directional broadcasts replace the isotropic ones and h replaces k, all the results that will be presented for coronas hold also for sectors. Therefore, sector training will not be further discussed and we shall focus on corona training only.

```
Procedure Actor (k);
  t := 0;
  repeat
  for i := k − 1 downto 0 do
      transmit beacon i up to corona C_{k−1};
      transmit beacon i up to corona C_i;
  t := t + 2k;
  until t > τ
```

Fig. 6. *Illustrating the actor protocol.*

C. The Sensor Behavior

In order to describe the protocol for the sensors, it is crucial to point out that the sensors are aware of the actor behavior and of the number of coronas k, which they can learn from the control-broadcast beacons.

We begin by sketching the behavior of a generic sensor, regardless of its type. To determine its corona, a sensor uses two $(\lfloor \log k \rfloor + 1)$-bit registers, named *min* and *max*. At any instant, the *min* (*max*) register keeps track of the largest (smallest) corona identity, heard so far via a control-broadcast (data-broadcast), smaller than (larger than or equal to) the corona to which the sensor belongs. The *min* and *max* registers are initialized to -1 and $k-1$, respectively, because initially each sensor can belong to any corona in $[0, \ldots, k-1]$.

From now on, the interval $[min+1, \ldots, max]$ is called the *corona identity range*, and its width $max - min$ is denoted by λ. From the above discussion, the following *training condition* is verified:

Lemma III.1. *A sensor which belongs to corona c, with $c \geq 0$, is trained when $\max = c$ and $\min = c-1$, and hence $\lambda = 1$.*

We assume that each sensor wakes up *at random* between the 0-th and the $2(k-1)$-st time slot and starts listening to the actor for d time slots, with $d \geq 2$. During its awake period, the sensor properly sets the min and max registers according to the actor's transmissions received. After its first awake period, each sensor guesses to belong to corona $\lceil \frac{\min + \max}{2} \rceil$ and goes to sleep until the actor transmits such a corona identity. At its next awakening, if the sensor receives the data-broadcast relative to the corona identity it guessed, its corona identity range becomes $[C_{\min+1}, \ldots, C_{\lceil \frac{\min+\max}{2} \rceil}]$, otherwise the corona range becomes $[C_{\lceil \frac{\min+\max}{2} \rceil + 1}, \ldots, C_{\max}]$. Such a binary search continues until the range boundaries differ by one, and thus the sensor is trained.

The details are spelled out in Figure 7. A sensor listens for an awake period of d consecutive time slots. Since the sensor is asynchronous, it keeps track of two slots, one even and one odd, to understand whether it woke up at a data-broadcast or a control-broadcast. During the even slots, it stores in variable *first* either the beacon received, if any, or k (lines 4–7). During the odd slots, if the sensor does not receive any beacon, it is sure that it woke up at a control-broadcast. Thus, the actor is now data-broadcasting the beacon *first* and the corona of the sensor must be larger than $first$. In variable *control*, the sensor remembers the local time when the control-broadcast was received (lines 8–11).

On the other hand, if during the odd slots the sensor receives beacon c, three cases arise depending on what happened in the previous slot, namely, a control-broadcast was received (lines 13–15), a data-broadcast was received (lines 17–19), or a data-broadcast was not received (lines 21–24). The first

```
    Procedure Binary-Training (k, d);
1       trained := false; ν := t := 0; min := −1; max := k − 1;
2       while ¬ trained do
3         for i := 0 to d − 1 do
4             if even(i) then
5                 if received beacon c then
6                     first := c;
                  else
7                     first := k;
              else
8                 if ¬ received beacon c then
9                     if min ≤ first then
10                        min := first; update:=left;
11                        control:= t + i − 1;
                  else
                      cases
12                        c = first:
13                            if max ≥ c then
14                              max := c; update:=right;
15                              control:= t + i − 1;
16                        first≠ k and c = (first−1) mod k:
17                            if max ≥ first then
18                              max := first; update:=right;
19                              control:= t + i;
20                        first = k:
21                            if min ≤ (c + 1) mod k then
22                                min := (c + 1) mod k;
23                                update:=left;
24                                control:= t + i;
25            t := t + d − 1;
26        if max − min = 1 then
27            mycorona := max;
28            trained := true;
          else
29            guess:= ⌈(min + max)/2⌉;
30            alarm-clock := control + Wait();
31            sleep until alarm-clock rings;
```

Fig. 7. *Training protocol for a generic sensor.*

case is detected because the sensor hears the same beacon c twice, which implies that the sensor belongs to a corona whose identity is smaller than or equal to c. The second case happens when the sensor hears two distinct beacons differing by $1 \bmod k$, yielding that the sensor belongs to a corona smaller than or equal to *first*. The third case occurs when the sensor hears only the beacon c during the second slot, with the consequence that the sensor belongs to a corona larger than $(c+1) \bmod k$. At the end of the awake period, the sensor tests the training condition (lines 26–28). If it is not trained, by invoking the *Wait* procedure, the sensor intends to wake up again when the actor broadcasts the corona identity in the middle of the sensor corona identity range (lines 29–31). The time complexity of the *Binary-Training* protocol is $O(d)$ plus the time required for executing the *Wait* procedure.

So far, the behavior of the sensors during the Binary-Training task has been described independent of their type. Indeed, only the procedure *Wait*, which determines how long a sensor has to sleep in order to receive the beacon corresponding to the *guess* corona, depends on the sensor type. Such a procedure mainly influences the total time each sensor employs to be trained, and thus the total time τ of the training task.

In the following, the *Wait* procedures, one for free and one for periodic sensors, are specified and analyzed.

1) Free Sensor Behavior: This subsection deals with sensors that can freely choose their awakening time. So they set the alarm clock when, according to their local time, the actor transmits the *guess* corona identity.

The *Wait* procedure is outlined in Figure 8. The sensor

```
    Function Wait: integer;
1     if update = right then
2         Wait := 2⌊(max−min)/2⌋;
      else
3         Wait := 2 (k − ⌊(max − min)/2⌋);
```

Fig. 8. *The* Wait *procedure invoked for the free sensors.*

sleeps for an interval which depends on the *guess* corona and the last modified boundary of the corona identity range.

Consider a sensor that finishes its current awake period and invokes the *Wait* procedure. If *update=right*, then *max* is the beacon transmitted via a control-broadcast at time slot *control* (see Figure 7). Since the *guess* corona is smaller than *max*, *guess* will be broadcast in the current actor cycle at time slot $control + 2\lfloor \frac{\max - \min}{2} \rfloor$. Whereas, if *update=left*, then *min* has been transmitted by the actor at the beginning of the current awake period. Since *guess* can only be larger than *min*, *guess* will be transmitted during the next actor cycle, at slot $control + 2(k - \lceil \frac{\max - \min}{2} \rceil)$. Clearly, the time complexity of the *Wait* procedure for the free sensors is $O(1)$.

Note that the sensor, setting the *control* variable, intends to wake up in the next period when the actor is transmitting the control-broadcast relative to the *guess* corona. However, the pseudo-code does not exploit this property. Indeed, the protocol works properly even if the sensor wakes up again when any data-broadcast or control-broadcast is transmitted. Moreover, since the sensor updates the *min* and *max* registers listening to the effective actor transmission, the sensor does not infer any information from its knowledge of the actor behavior, contrary to the previously known protocols [5], [6]. For all these reasons, the new protocol is robust to clock drift.

2) Periodic Sensor Behavior: In this subsection, the *Wait* procedure for the periodic sensors is devised. For the sake of the analysis, each sensor is assumed to wake up for the first time at a random instant $2s$, with $0 \leq s \leq k-1$. Recall that a sensor running this protocol always alternates d slots during which it is awake and $L - d$ slots in which it sleeps. In each of the d slots where the sensor is awake, it updates its position according to the heard data. At each awakening, each sensor hears groups of $\frac{d}{2}$ consecutive corona identities, broadcast by the actor. Since two consecutive awake periods start L time slots apart, the corresponding first beacons transmitted by the actor are $\frac{L}{2} \bmod k$ apart. Hence, a periodic sensor which does not skip any awake period hears the k corona identities in a specific order which depends on the parameters d, L, and on the time slot s at which the sensor wakes up for the first time. On the top of such an order, the Binary-Training protocol imposes the binary search scheme on the corona identity range by means of the *Wait* procedure, which forces a sensor to skip awake periods until that in which *guess* is transmitted.

The *Wait* procedure is given in Figure 9. Consider a sensor that finishes its current awake period at slot t and invokes the *Wait* procedure. At first, the sensor recomputes in variable *firstcorona* the beacon which was transmitted by the actor at the beginning of its current awake period. Indeed, if *update=right*, then max has been updated at each time slot and *firstcorona* is $(\max + \frac{d}{2}) \bmod k$. Whereas, if *update=left*, then min has been updated only at the first time slot of the

```
    Function Wait: integer;
1     if update = right then
2         firstcorona := (max + d/2) mod k;
      else
3         firstcorona := min;
4     γ := 1;
5     while guess ∉
      [(firstcorona − γ L/2 − d/2 + 1 mod k, (firstcorona − γ L/2) mod k] do
6         γ := γ + 1;
7     Wait := γL − d + 1;
```

Fig. 9. *The* Wait *procedure invoked for the periodic sensors.*

awake period, and *firstcorona* is exactly the corona identity stored in register *min* (lines 2-3). Note that in the first awake period, if two boundaries have been updated, register *update* must be equal to *right*. Thus, in a lookup process, the sensor checks during which subsequent awake period the *guess* beacon will be transmitted (lines 5–6), and stores in γ the number of awake periods to be skipped plus one. Indeed, since the corona identities transmitted at the beginning of two consecutive awake periods differ by $\frac{L}{2} \bmod k$, and $\frac{d}{2}$ beacons are transmitted in each awake period, the sensor knows which beacons it can receive in every awake period.

The time complexity of the Wait procedure, shown in Figure 9, is $O(\gamma d)$. However, such a complexity can be reduced by storing in each sensor a look-up table, as it will be shown at the end of Subsection IV-B.

IV. Correctness and Performance Analysis

In this section, the correctness and the performance of the Binary-Training protocol are discussed. The results proved in the next lemmas hold for both free and periodic sensors.

Lemma IV.1. *Each sensor requires at least* 2 *consecutive time slots to learn its relative position with respect to the beacon transmitted in the last data-broadcast.*

Proof. By contradiction, consider a sensor that listens to the actor for just one slot. If the sensor receives beacon c, it cannot distinguish whether it hears a control- or a data-broadcast. On the other hand, if the sensor does not receive any beacon, although it is aware that the actor transmits a data-broadcast, it cannot update the *min* register because it does not know the transmitted beacon. Therefore, in both cases the sensor cannot update its corona identity range. Consider now a sensor that has listened for two consecutive time slots. Since $i = 1$, the sensor executes the code in lines 8–24, and hence it sets either min or max learning its relative position with respect to the last data-broadcast beacon. □

As a consequence of Lemma IV.1, it is necessary that the length d of the awake period of both free and periodic sensors be at least 2 to allow all the sensors to be trained (such a condition is also sufficient only for free sensors, as it will be shown later). Let us now concentrate on how the width λ of the corona identity range decreases for any sensor. Precisely, in the first awake period of a sensor, λ reduces as follows:

Lemma IV.2. *Consider a sensor belonging to corona c that wakes up at time slot s, $0 \leq s \leq 2k-1$, when the actor transmits beacon K_s, with $0 \leq c, K_s \leq k-1$. If the sensor is untrained at the end of the first awake period, the width $\lambda = max - min$ of its corona identity range is:*

$$\lambda = \begin{cases} min\{k - K_s - 1, k - \frac{d}{2}\} & \text{if } c > K_s \\ K_s - \frac{d}{2} + 1 & \text{if } c \leq K_s \end{cases}$$

Proof. Consider the behavior of a sensor that at the end of its first awake period is still untrained. Assume that the sensor does not receive the data-broadcast transmitting beacon K_s, that is, $c > K_s$. If $K_s \geq \frac{d}{2}$, then the *min* boundary of its corona identity range is updated to K_s. Since the actor transmits at decreasing power levels, the next d transmissions will not update register *min*. Hence, the corona identity range becomes $[K_s + 1, \ldots, k-1]$. Whereas, if $K_s < \frac{d}{2} - 1$, the register *max* is updated because the sensor is awake while the actor transmits beacon $k-1$. However, overall $\frac{d}{2}$ coronas are excluded, leading to a corona range of width $k - \frac{d}{2}$. Assume now that the sensor receives the data-broadcast transmitting beacon K_s, that is, $c \leq K_s$. Then, the sensor updates the *max* boundary for $\frac{d}{2}$ times. Therefore, the new corona range becomes $[0, \ldots, K_s - \frac{d}{2}]$. Note that if $K_s < \frac{d}{2}$, the sensor will be trained. □

The following two results hold for trainable sensors, that is, for those sensors that after a finite time have $\lambda = 1$.

Lemma IV.3. *In each awake period but the first, every trainable sensor, which belongs to corona $c > 0$, updates only one boundary of its corona identity range unless it becomes trained. Every sensor in corona* 0*, always updates only one boundary.*

Proof. We proceed by contradiction; consider a sensor in corona $c > 0$ that updates both boundaries in the same awake period, but remains untrained. Let *min* and *max* be the values of the boundaries at the beginning of the awake period. During such an awake period, the sensor must have received the control-broadcast for a corona identity larger than *min* down to the data-broadcast for a corona identity smaller than *max* (passing through the control-broadcasts for corona identities 0 and $k-1$). However, this takes more than d time slots since, already at the end of the first awake period, at least $(min + 1) + (k - max) \geq \frac{d}{2}$ coronas are excluded by the corona identity range.

A sensor belonging to corona 0, whenever it wakes up, it will receives the actor's transmission. Thus, it sets *max* in each awake period. When it receives beacon 0, it is trained because $max - min = 0 - (-1) = 1$. □

As explained in Subsection III-C, in each awake period but the first, the width λ of the corona identity range is reduced by applying a binary search scheme on the interval $[min, \ldots, max]$ until $\lambda = 1$. This process requires a number of sleep/awake transitions, whose worst value is denoted by $\nu_{\max}$, bounded as follows:

Lemma IV.4. *A trainable sensor that belongs to corona c and wakes up for the first time at time slot s, $0 \leq s \leq 2k-1$, while the actor transmits beacon K_s, with $0 \leq c, K_s \leq k-1$,*

requires

$$\nu_{\max} \leq \begin{cases} 1+\lceil \log(\min\{k-K_s-1, k-\frac{d}{2}\})\rceil & \text{if } c > K_s \\ 1+\lceil \log(K_s-\frac{d}{2}+1)\rceil & \text{if } c \leq K_s \end{cases}$$

transitions to be trained.

Proof. After the first awake period, the corona identity range reduces by half at each awakening because the sensor learns its relative position with respect to the *guess* corona, which is in the middle of the corona identity range. Therefore, by Lemma IV.2, the result follows. □

It is worth noting that a free sensor is always trainable provided that $d \geq 2$ because, being free to set its alarm-clock, it is guaranteed to hear the beacon corresponding to the *guess* corona. In contrast, a periodic sensor is constrained in its awakenings and thus it is trainable only if some conditions on the parameters L, k and d are verified, as it will be proved in Subsection IV-B.

In order to analytically evaluate the performance of the *Binary-Training* protocol, in addition to ν_{max}, let ω_{max} be the worst overall awake time per sensor, and τ be the total time for training. Recalling that each awake period lasts for d time slots, one has $\omega_{max} = \nu_{max} d$. Note that τ measures the time required to terminate the whole training task for the actor, whereas each sensor counts in t its *local* training time, that is, how many slots elapse from its first wake up until it is trained. Hence, a sensor which is trained at local time t is trained at time $t+s$ for the actor, if s is the random time slot when the sensor wakes up the first time. Therefore, τ cannot be larger than $t_{max}+2k-1$, where t_{max} is the worst training time among the training times of all the sensors. The analysis of the total time required by Binary-Training depends on the *Wait* procedure, which determines how long a sensor has to sleep before receiving the beacon corresponding to the *guess* corona, and hence it is different for free and periodic sensors.

A. Free Sensors

In order to bound from above the total time τ for the training task, the following result is useful:

Lemma IV.5. *The training task for a free sensor that belongs to corona c cannot last more than $\tau_c = 2k(1+\lceil \log_2 c\rceil)$ time slots. Therefore, $\tau \leq 2k(1+\lceil \log_2 k\rceil)$.*

Proof. By applying the binary search scheme to the corona identity range, a sensor that belongs to corona c must exclude the coronas $0, 1, \ldots, c-1$ from its corona identity range by updating the register *min*. This can be done at most $\lceil \log_2 c\rceil$ times. Since the sensor waits at most $2k$ slots between two consecutive updates of *min*, the result follows. □

A consequence of the above lemma is that the inner coronas finish the training task earlier than the outer coronas. In this way, the wireless sensor network is formed from the center to the periphery. Hence, the performance of the Binary-Training protocol for free sensors can be summarized as follows:

Theorem IV.6. *All the free sensors are trainable if $d \geq 2$ and to be trained each free sensor requires $\nu_{\max} \leq 1+\lceil \log_2 k\rceil$, $\omega_{\max} = d\nu_{\max}$, and $\tau \leq 2k\nu_{\max}$.*

Proof. The proof follows from Lemmas IV.1, IV.4, and IV.5. □

B. Periodic Sensors

To analyze the performance of the Binary-Training protocol for periodic sensors, some properties on which beacons are received by the sensor, and in which order, are discussed. Denote with $\text{GCD}(a,b)$ the greatest common divisor between a and b, and let $L' = \frac{L}{2}$, $g = \text{GCD}(L',k)$, $d' = \frac{d}{2}$, $\hat{L}' = \frac{L'}{\text{GCD}(L',k)}$, and $\hat{k} = \frac{k}{\text{GCD}(L',k)}$. In order to derive the necessary and sufficient condition to train all the periodic sensors, the following observation is useful.

Lemma IV.7. *For fixed L, d, and k, assume that, during the first two slots, when the sensor wakes up for the first time, the actor has transmitted the data-broadcast K_s, with $0 \leq K_s \leq k-1$. Then the data-broadcast transmitted in the first two slots of the i-th sensor awake period is $(K_s - iL') \bmod k = \left(K_s - \text{GCD}(L',k)(i\hat{L}') \bmod \hat{k}\right) \bmod k$, assuming that the sensor does not skip any awake period. Overall only $\hat{k}$ different data-broadcasts can be transmitted by the actor in the first two slots of every sensor awake period, independent of how many awake periods the sensor performs. Such $\hat{k}$ data-broadcasts differ from each other by a multiple of $\text{GCD}(L',k)$.*

Proof. Consider a sensor for which, during its first awake period, the data-broadcast K_s has been the first one transmitted by the actor and which does not skip any awake period. The i-th awake period, $i \geq 0$, of such a sensor starts iL time slots later while the actor is data-broadcasting, during the first two slots of the sensor awake period, $(K_s - iL') \bmod k = (K_s - (iL') \bmod k) \bmod k$. Observe that L' and k can be rewritten as $L' = g\hat{L}'$ and $k = g\hat{k}$. Since $(iL') \bmod_k = g(i\hat{L}') \bmod \hat{k}$ (see [11]), $(iL') \bmod k$ is a multiple of g and generates only the $\hat{k}$ multiples of g in $[0, \ldots, k-1]$ while i varies in any interval of at least $\hat{k}$ consecutive integer values. Therefore, $(K_s - (iL') \bmod k) \bmod k = \left(K_s - g(i\hat{L}') \bmod \hat{k}\right) \bmod k$. Moreover, in any two awake periods, say the i-th and the j-th ones, such that $i > j$ and $i - j < \hat{k}$, the two first data-broadcasts transmitted are distinct and differ by a multiple of g. Whereas, the same first data-broadcasts are transmitted in any two awake periods i and j such that $i \equiv j \bmod \hat{k}$. □

For example, assume $L = 28$, $k = 8$, and $d = 14$, and consider a sensor that wakes up for the first time while the actor broadcasts $K_s = 0$. In Figure 10, a table A is depicted which shows in row i the corona identities heard at the i-th awake period. According to Lemma IV.7, A has $\hat{k} = \frac{8}{2} = 4$ rows and $d' = 7$ columns. For instance, column 0 shows the $\hat{k}$ different data-broadcasts $\{0, 2, 4, 6\}$ which can be transmitted in the first two slots of every sensor awake period and which differ by $g = \text{GCD}(14, 8) = 2$, while row 1 shows the 7 corona identities broadcast during its second awake period, assuming that the sensor does not skip it. Observe that the first beacon transmitted in this second awake period is $(K_s - iL') \bmod k = (0 - 1 \cdot 14) \bmod 8 = 2$. If the sensor does not skip any awake period, it wakes up in the next two awake periods while the

	0	1	2	3	4	5	6
0	0	7	6	5	4	3	2
1	2	1	0	7	6	5	4
2	4	3	2	1	0	7	6
3	6	5	4	3	2	1	0

Fig. 10. Table A showing the corona identities broadcast by the actor during the awake periods of a sensor, assuming it does not skip any awake period and that it woke up for the first time while the actor was transmitting $K_s = 0$.

actor transmits 4 and 6, respectively, as depicted in column 0. This behavior is periodic and in any subsequent awake period the sensor will wake up while the actor broadcasts one corona identity among $\{0, 2, 4, 6\}$.

As a consequence of Lemma IV.7, a sensor can hear, regardless of how long the training task lasts, $\hat{k}$ distinct sequences each of d' consecutive decreasing corona identities. If $d' < \text{GCD}(L', k)$, the sensor receives $\hat{k}$ non-overlapping sequences of corona identities, and hence only $\hat{k}d' < k$ corona identities. If $d' \geq \text{GCD}(L', k)$, the sensor hears at least once each of the k corona identities.

Lemma IV.8. *The training condition is satisfied for all the periodic sensors if and only if* $d' \geq \text{GCD}(L', k)$.

Proof. By Lemma IV.7, regardless of how long the training task lasts, a sensor can learn its relative position only respect to $\min\{k, d'\hat{k}\}$ different coronas even if it does not skip any awake period. Therefore, if $d' \geq \text{GCD}(L', k)$, since $\min\{k, d'\hat{k}\} = k$, for any *guess* corona of the Binary-Training protocol there is at least one of the subsequent $\hat{k}$ consecutive awake periods in which the sensor can hear *guess*. Whereas, if $d' < \text{GCD}(L', k)$, since $\min\{k, d'\hat{k}\} = d'\hat{k} < k$. there are corona identities which can never be heard by the sensor irrespective of the training task duration. If one of those corona identities is a *guess* corona for a sensor, the protocol cannot terminate for such a sensor, which thus remains untrained. □

Therefore, the performance of the Binary-Training protocol for periodic sensors is given by the following result.

Theorem IV.9. *For fixed* L, d, *and* k, *if* $d' < \text{GCD}(L', k)$ *then there are sensors which cannot be trained by the Binary-Training protocol; otherwise to be trained all the periodic sensors require* $\nu_{\max} \leq 1 + \lceil \log_2 k \rceil$, $\omega_{\max} = d\nu_{\max}$, *and* $\tau \leq \hat{k}L\nu_{\max}$.

Proof. The results for $\nu_{\max}$ and $\omega_{\max}$ follow from Lemma IV.4. With regard to τ, since the cycles of the actor and of the sensors last $2k$ and L slots, respectively, then the actor and the sensors are simultaneously at the beginning of their cycle every $\text{LCM}(2k, L) = \frac{2kL}{\text{GCD}(L,2k)} = \hat{k}L$ slots. In other words, the cycle of the actor-sensor system, i.e., the minimum time after which both the actor and a sensor are again in the initial condition, is of length $\hat{k}L$ slots. Since to hear each *guess* beacon a sensor has to wait at most a cycle of the actor-sensor system, and since at most $\nu_{\max}$ guesses are performed, the protocol takes $\tau \leq \hat{k}L\nu_{\max}$ time slots. □

However, taking into account the particular values that d can assume, better bounds on the performance parameters can be derived.

Theorem IV.10. *For fixed* L, d, *and* k, *one has:*

1) *if* $\text{GCD}(L', k) \leq d' < L' \bmod k$, *then* $\nu_{\max} \leq 1 + \lceil \log \frac{k}{d'} \rceil$, $\omega_{\max} = d\nu_{\max}$, *and* $\tau \leq \frac{k}{d'}L\nu_{\max}$;
2) *if* $L' \bmod k \leq d' < k$, *then* $\nu_{\max} \leq 1 + \lceil \log \frac{k}{d'} \rceil$, $\omega_{\max} = d\nu_{\max}$, *and* $\tau \leq \lceil \frac{k}{d'} \rceil L\nu_{\max}$;
3) *if* $d' = k$, *then* $\nu_{\max} = 1$ *and* $\omega_{\max} = \tau = d$.

Proof. The result trivially follows when $d' = \text{GCD}(L', K)$ because, by Lemma IV.7, the k coronas are partitioned into $\frac{k}{d'}$ non-overlapping intervals over which a binary search is performed to locate where *guess* is transmitted. Hence, the binary search takes $\nu_{\max} = 1 + \lceil \log \frac{k}{d'} \rceil$ guesses. Since each interval lasts $d = 2d'$ slots and since a sensor waits at most $\frac{k}{d'}L$ slots to hear each *guess* beacon, the results for $\omega_{\max}$ and τ follow.

When $d' = L' \bmod k$, if the sensor is awake for two consecutive awake periods, that is, for two awake periods starting at time slot t and $t + L$, it would hear $c - d' + 1$ as the last beacon of the first period and $c - d'$ as the first beacon of the second period, if c is the beacon heard at time t. Thus, the k corona identities are covered by $\lceil \frac{k}{d'} \rceil$ intervals (out of which $\lfloor \frac{k}{d'} \rfloor$ are non-overlapping) and a binary search is performed on such intervals to find where *guess* is transmitted. Since each interval lasts $d = 2d'$ slots and since a sensor waits at most $\lceil \frac{k}{d'} \rceil L$ slots to hear the *guess* corona identity, the bounds for $\omega_{\max}$ and τ hold.

When $d' = k$, the k corona identities are covered in a single interval, and each sensor is trained in the first awake period. Thus, the bounds are trivially derived.

Observe that when $\text{GCD}(L', k) < d' < L' \bmod k$ or $L' \bmod k < d' < k$, the number of intervals which cover the k corona identities cannot be greater than that in the case of $d' = \text{GCD}(L', k)$ and $d' = L' \bmod k$, respectively. Hence, the proof follows. □

With regard to the time complexity of the *Wait* procedure (Fig. 9), one can use a table T_{K_s}, where K_s is defined in Lemma IV.7, to faster compute γ. T_{K_s} consists of k rows and $\lceil \frac{d'}{g} \rceil$ columns. Given h and j, with $0 \leq h \leq k - 1$ and $0 \leq j \leq \lceil \frac{d'}{g} \rceil - 1$, $T_{K_s}(h, j)$ contains the awake period in which the sensor will hear the corona identity h between the start times jg and $(j+1)g$ of two consecutive awake periods. The value $T_{K_s}(h, j)$ verifies $0 \leq T_{K_s}(h, j) \leq \hat{k} - 1$ and it is intended as a relative position within the system actor-sensor cycle. In practice, row h of T_{K_s} contains all the awake periods in which the sensor can hear corona identity h during the system actor-sensor cycle if the sensor does not skip any awake period. It is worth noting that the same beacon can be heard by a sensor in more than one awake period (unless $d' = g$, in which case there is only a single column in T_{K_s}). Indeed, since each awake period includes d' consecutive corona identities and since distinct awake periods start with beacons which are multiples of g, beacon h is heard by a sensor in at most $\lceil \frac{d'}{g} \rceil$ awake periods, namely, for all those overlapping periods which include h.

Referring to the example in Figure 10, Figure 11 shows the content of T_0 for the same parameters, namely, $L = 28$, $k = 8$, and $d = 14$. For instance, row 5 of T_0 contains $T_0(5, 0) = 3$,

	0	1	2	3
0	0	1	2	3
1	1	2	3	∞
2	1	2	3	0
3	2	3	0	∞
4	2	3	0	1
5	3	0	1	∞
6	3	0	1	2
7	0	1	2	∞

Fig. 11. The table T_0 indicating the awake periods in which each corona identity is heard by a periodic sensor when $L = 28$, $k = 8$, and $d = 14$.

$T_0(5,1) = 0$, $T_0(5,2) = 1$, and $T_0(5,3) = \infty$, because beacon 5 is transmitted during the 3-rd awake period in one of the slots 0 and 1, during the 0-th awake period in one slot between 2 and 3, in the 1-st awake period in one slot between 4 and 5, while it is never transmitted in slot 6, as one can check in Figure 10.

To better understand how to build table T_{K_s}, imagine first constructing a table A_{K_s} by setting $A_{K_s}(u,v) = (K_s - uL' - v) \bmod k$. Since A_{K_s} contains the corona identities heard in each awake period by a sensor that wakes up for the first time while the actor broadcasts beacon K_s, one can derive the entries of T_{K_s} performing a kind of inverse computation. Precisely, if $A_{K_s}(u,v) = h$, with $0 \leq u \leq \hat{k} - 1$ and $0 \leq v \leq d' - 1$, then $T_{K_s}(h, \lfloor \frac{v}{g} \rfloor)$ is set to u. The unfilled entries in the last column of T_{K_s}, if any, are set to ∞. Clearly, this requires $O(\frac{kd'}{g})$ time and $O(\frac{kd'}{g} \log k)$ space for each sensor.

The above computation can be performed by each sensor at the beginning of the protocol, as soon as it knows its own K_s. Otherwise, such a computation can be done in a preprocessing phase, that is, before the sensor deployment, for a fixed value of K_s, like $K_s = 0$. When $K_s \neq 0$, each entry of T_{K_s} can be derived by the sensor from the precomputed table T_0 as: $T_{K_s}(h,j) = T_0((h - K_s) \bmod k, j)$. In other words, T_{K_s} corresponds to a row cyclic shift of T_0.

Finally, the number γ required in the *Wait* procedure of Figure 9 is obtained in $O(\frac{d'}{g})$ time by computing

$$\gamma = \min_{0 \leq j \leq \lceil \frac{d'}{g} \rceil - 1} \{\gamma_j : \gamma_j > 0\}$$

where

$$\gamma_j = (T_{K_s}(guess, j) - T_{K_s}(firstcorona, 0)) \bmod \hat{k}.$$

In fact, one computes the minimum number γ_j of the awake periods between each occurrence of the *guess* corona, $T_{K_s}(guess, j)$, in the system actor-sensor cycle and the current awake period, given by $T_{K_s}(firstcorona, 0)$.

For example, consider a sensor with $K_s = 0$, which has $guess = 5$ and $firstcorona = 2$. Since $T_0(2,0) = 1$, one has $\gamma_0 = (T_0(5,0) - 1) \bmod 4 = 2$, $\gamma_1 = (T_0(5,1) - 1) \bmod 4 = 3$, $\gamma_2 = (T_0(5,2) - 1) \bmod 4 = 0$, and $\gamma_3 = (\infty - 1) \bmod 4 = \infty$. Hence, $\gamma = \min\{2, 3, \infty\} = 2$, and the sensor has to wait $2L - d + 1 = 56 - 14 + 1 = 43$ slots.

C. Energy Consumption

In this subsection, the energy required for the Binary-Training protocol is evaluated under a realistic estimate of the power consumed by the sensors in their different operative modes.

During the training task, when a sensor is awake, its CPU is active and its radio is listening or receiving. In contrast, when a sensor is sleeping, its CPU is not active, its timer is on, and its radio is off. Let e_{a} and e_{s} be the energy consumed during a time slot by a sensor when it is listening/receiving or sleeping, respectively. Since the radio startup and shutdown require a non negligible overhead, let e_{t} denote the energy consumed for a sleep/awake transition followed by an awake/sleep transition. Thus, denoted with ν and ω, respectively, the number of wake/sleep transitions and the overall awake time, the total energy E depleted by a sensor is:

$$E = \nu e_{\mathrm{t}} + \omega e_{\mathrm{a}} + (\tau - \omega) e_{\mathrm{s}} \tag{1}$$

An upper bound on the energy drained by the training protocol for a free sensor is obtained from Equation 1 by substituting the worst case bounds for ν, ω, and τ given in Theorem IV.6, thus having:

$$E < (1 + \lceil \log k \rceil)(e_{\mathrm{t}} + d e_{\mathrm{a}} + 2k e_{\mathrm{s}})$$

Similarly, the energy spent by the protocol for periodic sensors is derived from Equation 1 by using the upper bounds provided in Theorem IV.10, observing that $d' \geq \mathrm{GCD}(L', k)$:

$$E < \left(1 + \log \left\lceil \frac{k}{\mathrm{GCD}\left(\frac{L}{2}, k\right)} \right\rceil\right) \left(e_{\mathrm{t}} + d e_{\mathrm{a}} + \frac{kL}{\mathrm{GCD}(\frac{L}{2}, k)} e_{\mathrm{s}}\right)$$

In order to evaluate the energy drained in a realistic setting, Table I reports the power consumed by a sensor in different operational modes. The data refer to the TinyNode 584, produced by Shockfish S.A., and are the customary values for the smallest sensors one can buy [8]. The sensors have as a power source two customary 1.2 Volt batteries, with a capacity of 1900 mAh each, and hence they have an energy supply of 4.56 Joule. As one can check in Table I, listening is nearly as expensive as receiving. The radio startup and shutdown require a power consumption, which cannot be higher than that in the active mode, and they take a non negligible amount of time (about 1 ms each). The above constraint influences the behavior of the protocol because it gives a lower bound on the the length of the sensor sleep period, which must be sufficient to allow both radio startup and shutdown, and thus cannot be shorter than 2 ms. Hence, a time slot of 2 ms is utilized. Note that such a slot duration is enough to accommodate within it the $O(\frac{d'}{g})$ computation time required in the worst case by a periodic sensor. In summary, from the data of Table I, one has that $e_{\mathrm{t}} = 30 \cdot 1 + 30 \cdot 1 = 60$ μJ, $e_{\mathrm{s}} = 0.015 \cdot 2 = 0.030$ μJ, and $e_{\mathrm{a}} = 32 \cdot 2 = 64$ μJ.

It is easy to see that since the actual value of e_{s} is negligible with respect to e_{t} and e_{a}, which in turn are comparable, the periodic sensors, which require a smaller $\nu_{\max}$, consume slightly less energy than the free ones, which in turn are trained faster.

V. Experimental Tests

In this section, the worst case and average case performance of the Binary-Training protocol are experimentally tested and

TABLE I

ESTIMATE OF SENSOR POWER CONSUMPTION IN DIFFERENT OPERATIONAL MODES AT 2.5 VOLT.

Sensor Mode	Current Draw	Power Consume
CPU inactive, timer on, radio off	6 μA	0.015 mW
CPU switch on, radio startup	3 mA	< 30 mW
CPU switch off, radio shutdown	3 mA	< 30 mW
CPU active, radio listening or RX	12 mA	32 mW

compared with the asynchronous corona training protocols previously presented in [5]. The algorithms were written in C++ and the experiments were run on an AMD Athlon X2 4800+ with 2 GB RAM. Since in the heterogeneous networks the free sensors do not influence the performance of the periodic ones, and vice versa, the Binary-Training protocol has been tested training either only free or only periodic sensors. In this way, the comparison with the previous protocols, which deal only with homogeneous networks, is more evident. In this section, the protocol for free or for periodic sensors is called BinFree and BinPeriodic, respectively.

In the simulation, there are $N = 10000$ sensors uniformly and randomly distributed within a circle of radius ρ, centered at the actor and inscribed in a square. Precisely, the Cartesian coordinates of each sensor are randomly generated by choosing two real numbers uniformly distributed in the range $[-\rho, \rho]$. The generation proceeds until N sensors are placed inside the circle, thus discarding those laying outside. Then, for each sensor, its first wake up time is randomly generated by choosing an integer number uniformly distributed in the range $[0, k-1]$.

Consider first some experiments comparing the performance of the BinFree protocol versus the BinPeriodic one. In the simulations reported in Figures 12-16, the number k of coronas is 64 and the length L of the sensor sleep-awake cycle is 216. The choice of k depends on the ratio between the full transmission range ρ of the actor isotropic antenna and the width of a corona, which in turn is assumed to be equal to the sensor transmission range r. Since practical orders of magnitude for ρ and r are hundreds and tens meters [8], our simulation assumes $\rho = 640$ and $r = 10$ meters, respectively. With regards to L, L is selected larger than k in order to allow d to span all possible values still maintaining a reasonably long sleep period $L - d$. Since the BinPeriodic protocol trains all the sensors only if $\frac{d}{2} \geq \text{GCD}(\frac{L}{2}, k) = \text{GCD}(108, 64) = 4$, the sensor awake period d varies between $2\text{GCD}(\frac{L}{2}, k) = 8$ and $2k = 128$ with a step of 8. The results are averaged over 3 independent experiments, which only differ in the deployment distribution of the sensors and in the sensor first wakeup times. If the network is dense enough to guarantee that in each corona there is at least one sensor for each first wakeup time, then $\nu_{\max}$ does not depend on the network density. Thus, $\nu_{\max}$ is always the same in different experiments, while the average number of transitions, denoted by ν_{avg}, may slightly change depending on the sensor first wakeup time distribution.

Figure 12 shows the number of transitions for the different values of d. According to Theorems IV.4 and IV.9, when

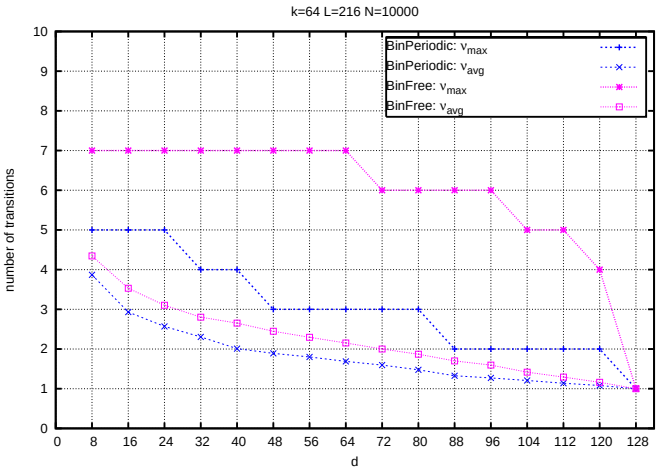

Fig. 12. *Number of transitions when* $k = 64$, $L = 216$, *and* $8 \leq d \leq 128$.

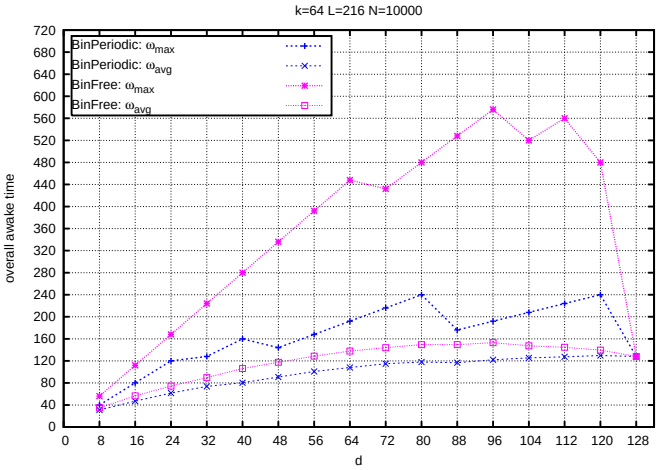

Fig. 13. *Overall sensor awake time slots when* $k = 64$, $L = 216$, *and* $8 \leq d \leq 128$.

$d = 2\text{GCD}(L', k) = 8$, BinFree and BinPeriodic have $\nu_{\max} = 1 + \lceil \log(k - \frac{d}{2}) \rceil = 7$ and $\nu_{\max} = 1 + \lceil \log(\frac{k}{\frac{d}{2}}) \rceil = 5$, respectively. Similarly, when $d = 2L' \bmod k = 88$, BinFree and BinPeriodic require $\nu_{\max} = 6$ and $\nu_{\max} = 2$, respectively. Clearly, increasing d, the gain of BinPeriodic over BinFree increases. With regard to average performance, although one notes that ν_{avg} considerably improves over $\nu_{\max}$ for both protocols, the improvement is higher for BinFree.

Figure 13 presents $\omega_{\max} = \nu_{\max} d$ and $\omega_{\text{avg}} = \nu_{\text{avg}} d$, which measure, respectively, the worst and average overall awake time spent by each sensor to be trained. Clearly, BinFree exhibits awake times longer than those of BinPeriodic since it requires a larger number of transitions. Although the number of transitions decreases as d increases, Figure 13 illustrates that the average overall awake time is slightly increasing for both protocols, except when d approaches $2k$, when all protocols take $\omega = 2k$. It is worthy to note that BinFree can train all the sensors even when $d = 2$, and in that case it achieves the absolute minimum for $\omega_{\max} = 2\nu_{\max} = 14$.

Figure 14 exhibits the total time τ required to accomplish the BinFree protocol for all the sensors in corona $c = 2^i$, with $0 \leq i \leq 6$, when $k = 64$, and either $d = 32$ or

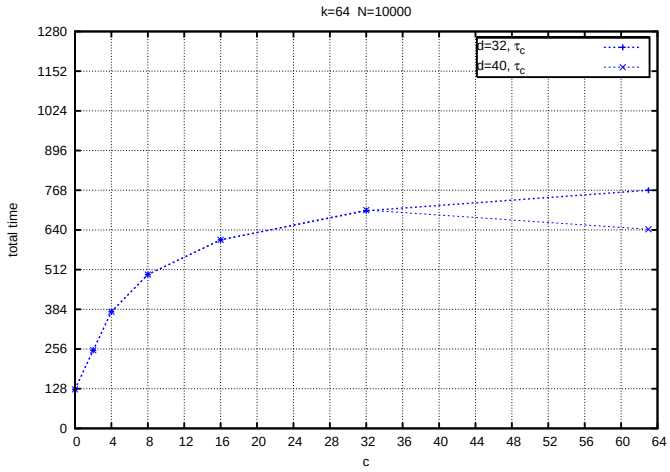

Fig. 14. *Total time slots required by the BinFree protocol to train all the sensors in corona* $c = 2^i$, *with* $0 \leq i \leq 6$, *when* $k = 64$.

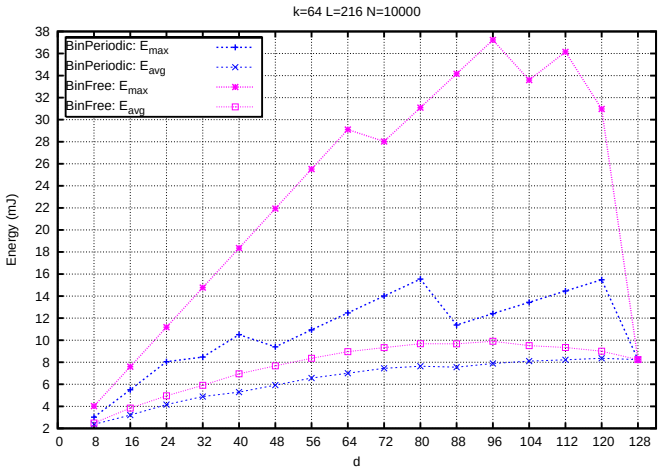

Fig. 16. *Energy consumed when* $k = 64$, $L = 216$, *and* $8 \leq d \leq 128$.

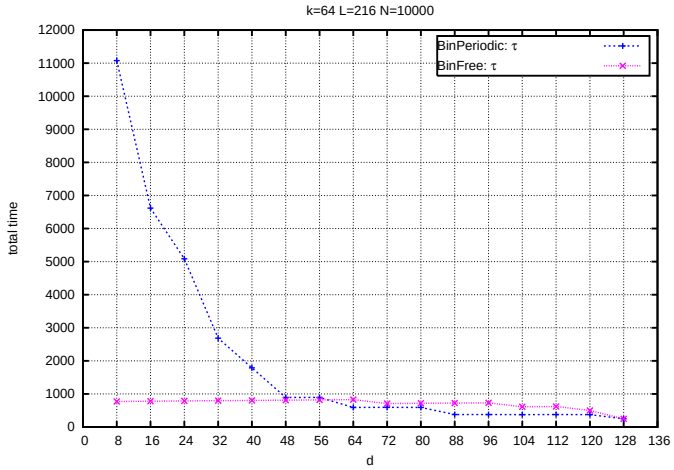

Fig. 15. *Total time slots when* $k = 64$, $L = 216$, *and* $8 \leq d \leq 128$.

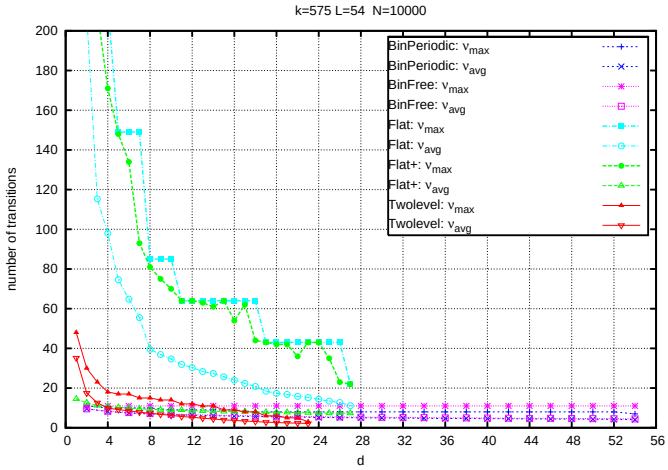

Fig. 17. *Number of transitions when* $k = 575$, $L = 54$, *and* $1 \leq d \leq 54$.

$d = 40$. The graphic confirms the results for the total time τ_c given in Lemma IV.5, that is $\tau \leq 2k(1 + \lceil \log_2 c \rceil)$. Figure 15 shows the total time τ required by the two protocols to train all the sensors in the network. BinPeriodic requires a total time extremely larger than that of BinFree when $d = 2\text{GCD}(\frac{L}{2}, k) = 8$. In fact, for such a value of d, to receive the beacon corresponding to the guessed corona, a free sensor has to wait at most $2k$ slots for each transition, whereas a periodic sensor has to wait at most $\frac{kL}{\text{GCD}(L',k)}$ slots, that is a cycle of the actor-sensor system. The total time of the BinPeriodic protocol neatly decreases when d increases until it becomes comparable with that of BinFree for $d \geq \frac{k}{2}$. Indeed, when d is sufficiently large the corona identities transmitted in different awake periods overlap. Hence, the same corona identity can be received by the periodic sensor during several awake periods of the same actor-sensor cycle, and in general, the sensor waits much less than $\frac{kL}{\text{GCD}(L',k)}$ slots to receive the beacon corresponding to the guessed corona. Note that the total time also decreases because, when d increases, the number of transitions required to train a sensor decreases.

Figure 16 shows the energy consumed by a sensor in the worst and average cases, denoted by $E_{\max}$ and E_{avg}, respectively, for both the BinPeriodic and BinFree protocols, where the time slot length is set to 2 ms. It is worth noting that the graphic of the energy has the same profile as that of the overall awake time. In fact, since the actual value of e_{s} is negligible with respect to e_{t} and e_{a}, which in turn are comparable, the energy grows proportionally to $\nu(d+1)$. Therefore, although BinPeriodic has a higher τ than BinFree when $8 \leq d \leq 48$, the former always consumes less energy than the latter. In the worst case, the energy depleted by the Binary-Training protocol is 38 mJ. Since the energy supplied by a sensor is about 4.56 J, the whole training task consumes at most $8/1000$ of the entire energy budget.

In conclusion, a heterogeneous wireless sensor network should use smaller values of d for the free sensors and larger values of d for the periodic sensors. In this way, the BinFree protocol optimizes the overall awake time and the energy consumed, without substantially penalizing the number of transitions, whereas the BinPeriodic protocol optimizes the number of sleep/awake transitions slightly increasing the overall awake time and the energy consumption.

Consider now some experiments where the new Binary-Training protocol is compared with the Flat, Flat+, and TwoLevel protocols, proposed in [5], for homogeneous networks of periodic sensors.

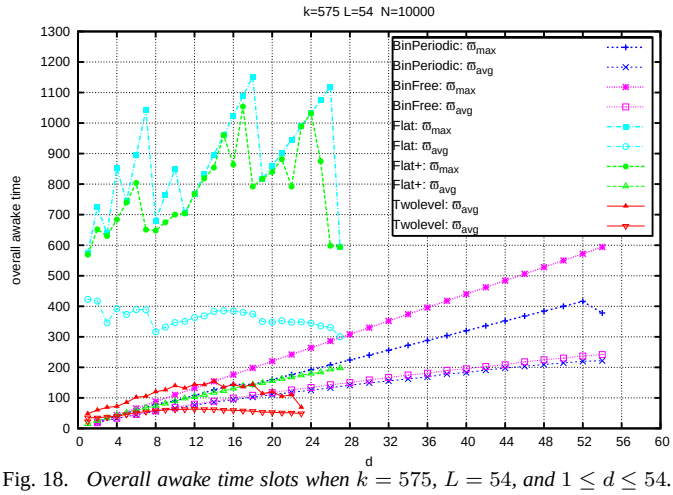

Fig. 18. *Overall awake time slots when* $k = 575$, $L = 54$, *and* $1 \leq d \leq 54$.

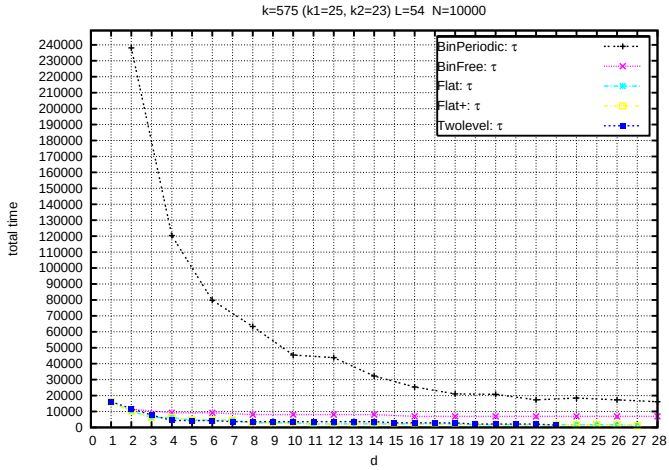

Fig. 19. *Total time slots when* $k = 575$, $L = 54$, *and* $1 \leq d \leq 54$.

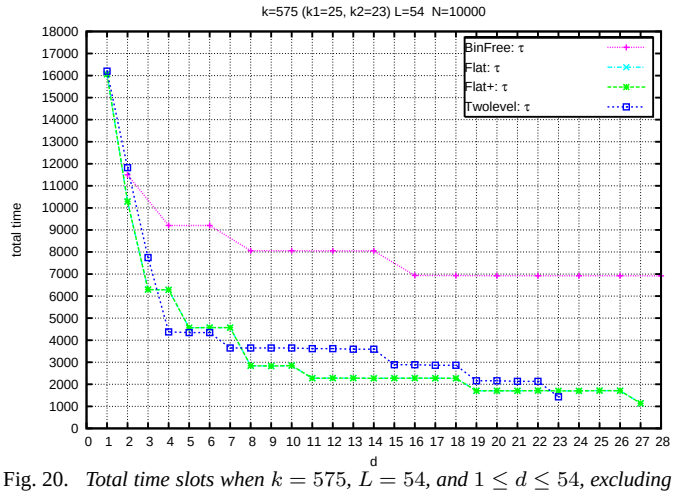

Fig. 20. *Total time slots when* $k = 575$, $L = 54$, *and* $1 \leq d \leq 54$, *excluding BinPeriodic.*

In the simulations reported in Figures 17-21, the number k of coronas is fixed to 575, the length L of the sensor sleep-awake cycle is 54 and the sensor awake period d varies between 1 and 54 with a step of 4. The numbers of macro-coronas and micro-coronas for TwoLevel are, respectively, $k_1 = 25$ and $k_2 = 23$, which indeed give $k = k_1 * k_2 = 575$. Note that d is bounded by the length L of the sensor cycle, while for $d = 1$, only the previously known algorithms are defined. In fact, according to Lemma IV.1, Binary-Training requires at least 2 consecutive slots to learn something.

The experiments show how both BinFree and BinPeriodic outperform Flat and Flat+ with respect to $\nu_{\max}$ and ν_{avg} (Figure 17), and to $\omega_{\max}$ and ω_{avg} (Figure 18). In particular, for ν_{avg}, although the corona identity range is guaranteed to decrease at each awakening applying either Flat+ or Binary-Training, its range decreases faster using Binary-Training. Indeed, this last protocol halves the corona identity range at each awakening of the sensor. With regard to TwoLevel, its number of transitions is smaller than that of Binary-Training only when d is approximately the same as the number of macro- and micro-coronas. Indeed, when $d = 23$, TwoLevel can train the sensors in just 3 transitions, whereas Binary-Training still uses a logarithmic number of transitions. Clearly, a similar observation holds for the overall sensor awake time.

Concerning τ, Figure 19 shows that the new protocol for periodic sensors is worse than the previous ones when d is very small, confirming that periodic sensors benefit from a moderately long awake period. One can note that, according to Theorems IV.6 and IV.10, BinFree and BinPeriodic have a total time bounded by their number of transitions multiplied by twice the number of coronas and by the Flat total time, respectively. As shown in Figure 20, BinFree has about a double total time with respect to all the protocols (but BinPeriodic) because $BinFree$ uses both data- and control-broadcasts, and hence in d time slots it hears $\frac{d}{2}$ corona identities, while the others hear d corona identities. However, the larger time of BinFree is widely counterbalanced by its much lower number of transitions which lead to a moderate energy consumption (see Figure 21). Indeed, BinPeriodic depletes the minimum amount of energy, in both the worst and average cases, with respect to all protocols but TwoLevel. Although TwoLevel has the minimum energy consumption in the average case, it requires a specific actor behavior [5] different from that used by all the other protocols.

The comparison between Flat and Binary-Training for periodic sensors reveals the bicriteria optimization behind a training task: one can either minimize the energy consumption or speed up the training task. Moreover, it is worth noting that in both Flat and Flat+, when the actor transmission is not received, the sensors update the corona identity range deriving from their local time the beacon transmitted by the actor. This makes the Flat and Flat+ protocols very sensitive to clock drift.

Finally, the above experiments show that Binary-Training for free sensors offers, especially for small values of d, the best compromise for both optimization criteria. Hence, the heterogeneous network takes advantage of the free sensors to become quickly operative, and of the periodic sensors to increase its longevity.

VI. Concluding Remarks and Open Questions

In this paper we have proposed an energy-efficient location training protocol for heterogeneous actor-centric sensor networks where the sensors acquire coarse-grain location

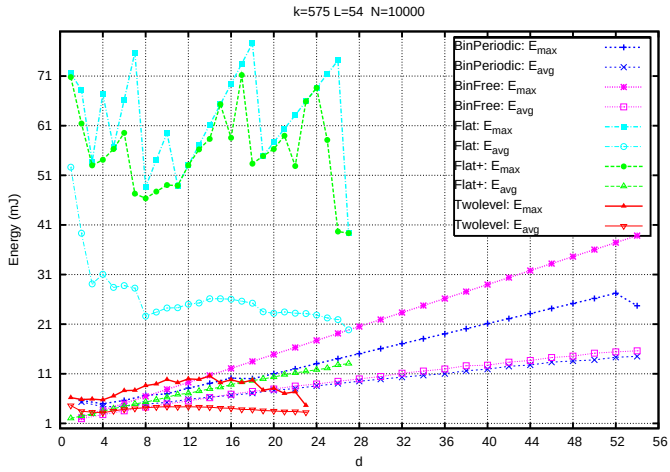

Fig. 21. *Energy consumed when* $k = 575$, $L = 54$, *and* $1 \leq d \leq 54$.

awareness with respect to the actor in their vicinity. The sensors differ in their ability to dynamically alter their sleep times: the *periodic* sensors feature sleep periods of predefined lengths, established at fabrication time; the *free* sensors can dynamically change their sleep times, under program control.

Our analytical analysis, confirmed by experimental evaluation, has shown that the proposed protocol outperforms the best previously-known training protocols in terms of number of sleep/awake transitions, overall awake time, and energy consumption.

Our experimental studies have suggested practical choices for the length d of the awake periods: smaller values of d for the free sensors and larger values of d for the periodic ones.

An interesting open question is whether the energy consumption of the protocol is optimal or not. To derive a lower bound, one needs a computational model, which is difficult to be formally defined because it depends on the actor transmission behavior, on how much information is transmitted each time, and on whether sensors are awake or not. Nonetheless, a lower bound can be derived when $d = 2$ because in this case the awake period has the minimal length required to perform one comparison. Since the actor transmits beacons in sorted order and a sensor has to search its corona among k beacons, at least $\nu_{max} \geq \log k$ transitions are needed. Thus the energy consumption of our protocol is optimal. In general, when d is an arbitrary function of k, such an argument does not apply and deriving a lower bound on the energy consumption remains an open question. as a future work, we also intend to compare the Binary-Training protocol with variants of the synchronous training algorithms, proposed in [6], properly modified so as to tolerate clock drift. Finally, the proposed protocol assumes that each sensor receives beacons from only one actor. When multiple actors are in close proximity, the protocol can still be used provided that the actors communicate among them in order to properly set their positions so as to avoid conflicts. Nonetheless, optimal actor placement is an interesting problem that deserves further investigation.

ACKNOWLEDGMENT

The authors thank D. Borgia for writing the C++ code used in the simulations. Prof. Olariu is funded by the US National Science Foundation under grant CNS-0721563.

REFERENCES

[1] S. Ainsworth, H. AbdelSalam and S. Rizvi, Towards maximum longevity of energy limited SANET, *Proc. ACM SANET*, Montreal, Canada, 2007.
[2] I.F. Akyildiz and I. Kasimoglu, Wireless sensor and actor networks: Research challenges, *Ad Hoc Networks*, 2, 351–367, 2004.
[3] I.F. Akyildiz, W. Su, Y. Sankarasubramanian and E. Cayirci, Wireless sensor networks: A survey, *Computer Networks*, 38(4), 393–422, 2002.
[4] J. Bachrach and C. Taylor, *Handbook of Sensor Networks*, Wiley, 2005.
[5] F. Barsi, A. A. Bertossi, F. Betti Sorbelli, R. Ciotti, S. Olariu and M.C. Pinotti, Asynchronous corona training protocols in wireless sensor and actor networks, *IEEE Transactions on Parallel and Distributed Systems*, 20(8), 1216–1230, 2009.
[6] A. A. Bertossi, S. Olariu and M.C. Pinotti, Efficient corona training protocols for sensor networks, *Theoretical Computer Science*, 402(1), 2–15, 2008.
[7] N. Bulusu, J. Heidemann, D. Estrin and T. Tran, Self-configuring localization systems: Design and experimental evaluation, *ACM Transactions on Embedded Computing Systems*, 3(1), 24–60, 2004.
[8] N. Burri, P. von Rickenbach and R. Wattenhofer, Dozer: Ultra-low power data gathering in sensor networks, *Proc. IPSN'07*, Cambridge, MA, April 2007.
[9] M. Eltoweissy, D. Gracanin, S. Olariu and M. F. Younis, Agile sensor network systems, *Ad Hoc and Sensor Wireless Networks*, 4(1-2), 97–124, 2007.
[10] S. Ghiasi, A. Srivastava, X. Yang and M. Sarrafzadeh, Optimal energy-aware clustering in sensor networks, *Sensors*, 2, 258–269, 2002.
[11] H. Griffin, *Elementary Theory of Numbers*, McGraw Hill, New York, 1954.
[12] T. He, C. Huang, B.M. Blum, J.A. Stankovic and T. Abdelzaher, Range-free localization schemes for large scale sensor networks, *Proc. ACM MobiCom*, San Diego, CA, September 2003.
[13] K. Langendoen and N. Reijers, Distributed localization algorithms, *Embedded Systems Handbook*, R. Zurawski (Editor), CRC Press, 2004.
[14] J. Ma, W. Lou and X.-Y. Li, Energy efficient TDMA sleep scheduling in wireless sensor networks, *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, 2009.
[15] T. Melodia, D. Pompili, V.K. Gungor and I.F. Akyildiz, Communication and coordination in wireless sensor and actor networks, *IEEE Transactions on Mobile Computing*, 6(10), 1116–1129, 2007.
[16] A. Navarra and A. Tofani, Distributed localization strategies for sensor networks, *Proc. 4th IEEE MASS*, Pisa, Italy, October 2007.
[17] D. Nicolescu, Positioning in ad-hoc sensor networks, *IEEE Network*, 18(4), 24–29, 2004.
[18] S. Olariu, M. Eltoweissy, and M. Younis, ANSWER: autonomous networks sensor systems, *Journal of Parallel and Distributed Computing*, 67, 114–126, 2007.
[19] S. Olariu and I. Stojmenovic, Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting, *Proc. IEEE INFOCOM*, Barcelona, Spain, 2006.
[20] N.B. Priyantha, H. Balakrishnan, E. Demaine and S. Teller, Anchor-free distributed localization in sensor networks, *Proc. ACM SenSys*, Los Angeles, CA, 2003.
[21] M. Rudafshani and S. Datta, Localization in wireless sensor networks, *Proc. IPSN '07*, Cambridge, MA, April 2007.
[22] A. Savvides, L. Girod, M. Srivastava and D. Estrin, Localization in sensor networks, *Wireless Sensor Networks*, C.S. Raghavendra, K.M. Sivalingam and T. Znati, Eds., Kluwer Academic, 2004.
[23] A. Wadaa, S. Olariu, L. Wilson, M. Eltoweissy and K. Jones, Training a wireless sensor network, *Mobile Networks and Applications*, 10(1), 151–168, 2005.
[24] J. Yick, D. Mukherjee and D. Ghosal, Wireless sensor network survey, *Computer Networks*, 52, 2292–2330, 2008.
[25] M. Youssef, M. Mah and A. Agrawala, Challenges: device-free passive localization for wireless environments, *Proc. ACM MobiCom*, Montreal, Canada September 2007
[26] Y. Zhu, S. Gortler and D. Thurston, Sensor network localization using sensor perturbation, *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, 2009.